Attached Appendix A

- Host and Network Monitoring:
  - Collection of Data via Operating System Calls for Minimal Intrusiveness
  - Common Data Formats Across Platforms
  - Distribution of Host and Network Statuses and Performance Histories
  - Discovery of Distributed Environment Configuration Changes
    - Detection of Host Failure
    - Detection of Host Startup
- Application-Level Instrumentation:
  - Low-Overhead Application API's
  - Common User-Specified Instrumentation Data Formats
  - Data Collection and Distribution Architecture
  - Grammar-Driven Event Correlation
- System Specifications:
  - Modeling of Application Systems
    - Structure, Capabilities, and Configuration
    - Requirements and Inter-Dependencies
  - Modeling of Hardware and Network Systems
    - Structure, Capabilities, and Configuration
  - Run-Time Access to Specification Information
    - Run-Time Loading of Specification Information
    - Run-Time Access via Object-Oriented API
- Resource Allocation Decision-Making:
  - Determination of Application-to-Host Mappings
  - Recovery from Hardware and Software Failures
  - Detection of and Recovery from Software Performance Problems
    - Control of Application Scalability
    - Reallocation of Applications to Hosts
  - Reallocation of Applications to Hosts based on Priority Changes
    - Application-to-Host Mappings for New Required Applications
    - Selection of Applications to be Shutdown
  - Resolution of Inter-Application Startup Dependencies
- Resource Control:
  - Startup, Shutdown, and Configuration of Distributed Applications
    - Interactive Operator Control via Operator Display
      - Creation of Defined System Configurations
      - Loading of Pre-defined System Configurations
      - Startup, Shutdown, and Configuration of Individual Applications
    - Automatic Control via Resource Manager Orders
    - Failure Detection Capabilities
      - Application Failure Detection via Interrupt Notification
      - Host Failure Detection via Internal Heartbeat Mechanism
- Displays / Visualization:
  - Host Configuration and Performance
  - Network Configuration and Performance
  - Application Software Performance
  - *Resource Allocation Decisions and State Information*
  - Software Status and Configuration
  - User-Configurable Instrumentation Display
  - Near Real-Time Display of Information
- Middleware
  - Reliable Message Passing
  - Location-Transparent TCP Client-Server Configuration
  - Automated Connections and Reconnections
    - Client and Server detection via UDP multicast
  - Many-to-Many Client-Server Connections supported
  - Message Callback Function Registration
  - TCP Connection Status Change Callback Function Registration

Attached Appendix B:

1) Event data message header:

```
long total_bytes              // total number of bytes in the event data message
long message_type             // message type designator *
char version[8]               // version of Instrumentation APIs *
char :est_name[24]            // test name for this event
double timetag                        // time stamp of when this event data message was sent
unsigned int gm_time          // GMT time stamp
unsigned int event_num        // event number
char process_name[24]         // name of application sending event data message
long pid                      // process id of application sending event data message
char host_name[64]            // host name that application is running on
long ip_addr                  // ip address of host
long tid                      // task id of application sending event data message
unsigned int thread_type      // thread type
unsigned int sequence_num     // sequence number of the event data message
double time_in_client         // time the API was called to create event data message
double time_server_received   // time the Instrumentation Daemon read in this event data message
double time_server_sent       // time the Instrumentation Daemon sent event data message to the Instrumentation Collector
```

2) The event data message format string contains data field names and format specifiers for each data field. The following data specifiers (borrowed from ANSI C) are supported:

%*r	: raw data, user defined, and number of bytes
%hi	: short signed 16 bit integer
%hd	: short signed 16 bit integer
%hu	: short unsigned 16 bit integer
%li	: long signed 32 bit integer
%ld	: long signed 32 bit integer
%lu	: long unsigned 32 bit integer
%lf	: IEEE double precision floating point - signed 64 bit floating point

%s	: null-terminated string data
%c	: character
%f	: IEEE single precision floating point - signed 32 bit floating point
%i	: signed 32 bit integer
%d	: signed 32 bit integer
%u	: unsigned 32 bit integer

Event string example: "StartTime %lf StopTime %lf TrackNumber %u Hostname %s"

3) The data fields are then packed using the MessageBuffer class described in Appendix A of the RMComms Middleware Design Report.

Attached Appendix C

Class	Description
TCPCommClient	RMComms client server communication client services: ▪ client configuration - client name, server port number, network interface to use (optional) ▪ connection and disconnection to servers - all servers, specific servers, or servers on specific hosts ▪ sending user-defined messages to connected servers - send to all servers or only to specific servers ▪ receiving user-defined messages from connected servers - registration of message handler callback functions for specific messages - polled or asynchronous message delivery ▪ monitoring of server connection statuses - *queries to determine connected server statuses* - notification of new server connections or broken server connections
TCPCommServer	RMComms client-server communication server services: ▪ server configuration - server name, server port number, network interface to use (optional) ▪ connection to new clients ▪ sending user-defined messages to connected clients - send to all clients or only to specific clients ▪ receiving user-defined messages from connected clients - registration of message handler callback functions for specific messages - *polled or asynchronous message delivery* ▪ monitoring of client connection statuses - queries to determine connected client statuses - notification of new client connections or broken client connections
TimeUtils	Clock access and time conversion services: ▪ read system clock time ▪ time conversions between GMT and local time ▪ time conversions to hours, minutes, seconds, day, month, year
SignalRegistry	User-defined signal (interrupt) handler registration services: ▪ register a signal handler function for a specified signal - invoked when interrupt occurs ▪ unregister a signal handler function for a specified signal

United States Patent & Trademark Office

Office of Initial Patent Examination -- Scanning Division

Application deficiencies found during scanning:

□ Page(s)__________of____________________________were not present for scanning.
(Document title)

□ Page(s)__________of____________________________were not present for scanning.
(Document title)

☑ ***Scanned copy is best available.*** Drawings.